Intercloud Control and Management Plane with XMPP

Peter Membrey*, Yuri Demchenko†
*Department of Computing
Faculty of Engineering, The Hong Kong Polytechnic University, Hong Kong SAR, China
Email: cspmembrey@comp.polyu.edu.hk
†System and Network Engineering Group
University of Amsterdam, Amsterdam, The Netherlands
Email: y.demchenko@uva.nl

***Abstract*—This paper introduces XMPP and suggests how this technology might be used to help implement Intercloud communication. It gives an introduction to XMPP and how the architecture fits together as well as a discussion of the services it provides 'out of the box'. It then discusses secondary benefits of the protocol and highlights how XMPP could be an appropriate base protocol for implementing the Intercloud Control and Management Plane. This is followed by discussion of early results from a research project that looks at the ease of extending XMPP and the tractability of the standardization process.**

I. Introduction

Applications today are designed for a very different environment from that of even a few years ago. Where previously applications would be designed to run on a single server or perhaps on a specialized cluster of servers, today applications are designed to be modular, distributed and to be composited in numerous ways. The age of the micro-service has arrived. These applications have the ability to dynamically adapt to changes in load and effectively self-heal.

These applications leverage Cloud architecture to give themselves an adaptive edge, but it is no longer enough to be able to burst within a single cloud. Applications are getting more complex with individual components requiring specific features that a single cloud may not be able to provide. Applications therefore need to be able to burst or even potentially migrate to a completely different cloud without prior agreement. Intercloud is a multi-layered technology and approach that will allow applications to operate natively across cloud platforms.

The Intercloud Architecture Framework (ICAF) [1] defines a general architectural framework for implementing multi-cloud services and introduces the Intercloud Control and Management Plane that defines a number of interfaces into the stack.

This paper explores some initial ideas with regards to using XMPP to implement the Intercloud Control and Management Plane layer in the Intercloud Architecture. First it discusses the XMPP protocol itself and then highlights some of the features that make it an ideal candidate for building an open standard for the ICCMP layer. Next is a brief discussion on an ongoing research project to determine how appropriate XMPP is for building such protocols and whether the standardization process is tractable. Finally it then discusses how these features can be used with reference to the ICCMP layer.

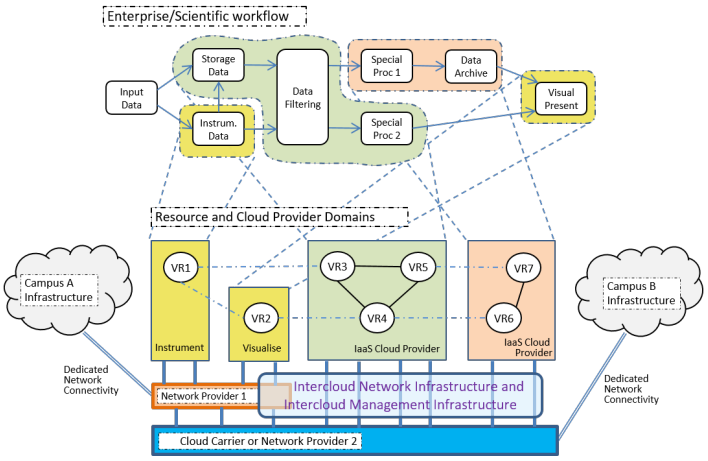

Fig. 1. Intercloud

The remainder of this paper is built as follows. Section II introduces some typical use cases and scenarios for multi-cloud applications that require tight integration. Section III introduces and gives an overview of two existing Intercloud paradigms (notably ICAF and IEEE P2302) that together form part of the proposed integrated solution. Section IV gives and overview of XMPP with Sections V, VI and VII discussing the services, benefits and applicability to Interlcoud respectively. Section VIII discusses a research project to demonstrate XMPP's extensibility. Lastly, Section IX presents the conclusions.

II. General use cases for Intercloud

The Intercloud Architecture has a number of relevant use cases. To provide additional context and overview, these use cases are provided:

1) For business and enterprise, migrating workloads to the cloud is seemingly vital for survival, especially in competitive markets. The first stage for such a migration would be to allow the integration of cloud services with

the existing legacy systems that a given entity may have. The second stage would be the migration of general services to the cloud and operating directly from there, although this itself would need to be staggered and done in stages. The final stage is for business and enterprise to develop and deploy cloud-native applications that integrate with the platform and feature scalability, reliability and self-healing attributes.

2) Infrastructure for cross-organization projects, especially those that are scientific in nature (where big data needs to be collected, distributed and processed) are becoming critical to deal with modern workloads. Such infrastructure is not limited to compute or storage and is likely to also consist of network and transport layer resources as well. This will need to be provisioned and managed on-demand [2].
3) Scalable disaster recovery is also a unique feature of the cloud environment. Elastic storage allows a business to store practically unlimited amounts of data, and pay on demand. This is ideal for backing up critical data or data that must be held for regulatory reasons. In addition, the ability to migrate a compute workload into the cloud, can provide a business with Business Continuity Planning (BCP) and Disaster Recovery Planning (DRP), which is also mandated by many regulators. This will allow the business to not only quickly restore operations but to potentially bring additional services online without requiring the business to restore its local systems first.

One of the defining features of cloud computing is that all of the resources available in the cloud can be provisioned on demand as needed. Resources such as compute, storage and networking should be provisionable and have the ability to integrate with the legacy systems that a business or entity may have. That is to say that there location of the cloud resources should not be visible or affect the operation of the legacy systems. In addition, the platform used by the cloud provider should not impose any restrictions on the architecture or use cases of the business or end user.

The primary goal of any IT infrastructure is to support the operations of the entity that built it. Enterprise systems and research infrastructure support commercial and scientific workloads respectively. The ability of the cloud to allow the provisioning and deploying of complex infrastructures on demand simplifies the building of such systems.

Figure 1 [3] gives an example workflow (either enterprise or scientific) and shows how cloud services can be used to closely map the workflow requirements to the infrastructure and then deployed on demand. It also demonstrates how an Intercloud architecture can support the sample use case. The entity in question has a number of legacy systems running on two inter-operating sites. In order to meet certain bandwidth and latency requirements (a guarantee of service, perhaps an SLA) the links between the two sites may need to be dedicated leased lines.

The example also demonstrates how the infrastructure takes advantage of different types of cloud computing, such as Platform as a Service (PaaS - VR6 to VR7), Infrastructure as a Service (IaaS - VR3 to VR5) and other virtualized resources (VR1 and VR2). These systems are able to integrate and interact with the legacy on-site systems described earlier.

To ensure the efficient operation of such an infrastructure, there is a requirement for management, both of the individual resources and the mechanisms by which these resources interact. Existing models and cloud platforms typically do not support this architecture and thus a new encompassing architecture, the Intercloud, is required in order to provide this functionality and allow the integration of existing cloud platforms.

III. Intercloud Frameworks

This section provides general information about two Intercloud frameworks that motivate the proposed XMPP based Intercloud signaling protocol. Although the frameworks are presented separately, they are not mutually exclusive and are complimentary in most respects.

A. ICAF and components

In order to create an Intercloud as described above, the necessary components can be defined as follows:

1) Multilayer Cloud Services Model (the CSM): provides a model similar to that of the OSI model and describes the vertical relationships (providing for interaction, integration and compatibility) between the existing cloud models (including IaaS, PaaS and SaaS). This component also encompasses any other layers or related components that form part of the cloud services infrastructure.

The layers that form the stack are defined as follows (where 'CMS' denotes CMS defined layers):

- CMS6: Customer owned and managed applications and resources
- CMS5: Intercloud Access and Deliver Infrastructure and components (ICADI) that provide the services and functions needed to connect multiple cloud domains
- CMS4: The Cloud Services Layer may contain any number of different cloud services (such as IaaS, PaaS and SaaS)
- CMS3: Orchestration and composition layer that is generally provided by Cloud Management Software such as OpenStack or OpenNebla
- CMS2: The Cloud Virtualization Layer, generally provided by platforms such as VMWare, Xen and KVM
- CMS1: The Physical Platform such as physical service hardware, network links and infrastructure)

2) Intercloud Control and Management Plane (ICCMP): provides a means of command and control for the Intercloud. This layer includes signaling (creating, managing and destroying sessions), sharing configuration (for migration), monitoring (determining the status of a resource) and real time optimization. This layer is also responsible for sharing the state needed for scaling an application or service and for routing data and resources as required.

3) Intercloud Federation Framework (ICFF): provides the protocols and processes need for clouds operating in independently administered domains to be able to federate. The layer should support this federation at each layer provided that the necessary services and gateways are available. Examples of the layers that should support this include both computational and business services and provide the means for exchanging name spaces and related semantics.

4) Intercloud Operation Framework (ICOF): provides high level functionality for inter-provider operational and business interaction. This includes the management of enterprise work flows, negotiation and management of SLAs (Service Level Agreements) and the related business accounting. A number of actors are defined by the ICOF layer along with how they relate to each other and their required interactions particularly in terms of management, ownership and resource operation. The ICOF layer is built upon and requires the support of both the ICCMP and ICFF layers.

B. Intercloud Control and Management Plane (ICCMP)

The Intercloud Control and Manage Plane (ICCMP - see Figure 2 [3], updated to show XMPP) provides the mechanism for cloud based applications to communicate, interact and manage sessions between each other and with common services provided by cloud platforms. This requires a reliable and scalable messaging layer that supports routing, service discovery and notification. This layer could be built directly on top of TCP/IP. However protocols already exist that offer these features with one such protocol being XMPP. XMPP automatically routes messages to endpoints and provides support for querying remote entities about the services, features and protocols that they support. In addition with rosters, XMPP provides a way to send push notifications directly to interested parties on a change of state.

C. IEEE P2302 Intercloud Interoperability and Integration Standard

The IEEE P2302 Intercloud model is highly inspired by the design of the Internet itself [4]. That is, just as the Internet is a 'ubiquitous and interoperable' network, so will the Intercloud be a 'ubiquitous and interoperable' network of clouds. Two key components that are introduced as part of this model are Intercloud Exchanges and the Intercloud root. Intercloud Exchanges are synonymous with the peering points as traditionally seen with ISPs. In this case an Intercloud Exchange is a nexus where clouds can inter-operate and communicate directly.

The Intercloud Root is envisioned to be similar in nature to DNS [5], a distributed entity that provides a trusted source of information regarding Intercloud Exchanges and that will act as an anchor point for various 'root' level services. The Intercloud Root is the subject of the research project discussed in Section VIII of this paper, where Raft was implemented over XMPP to provide distributed consensus.

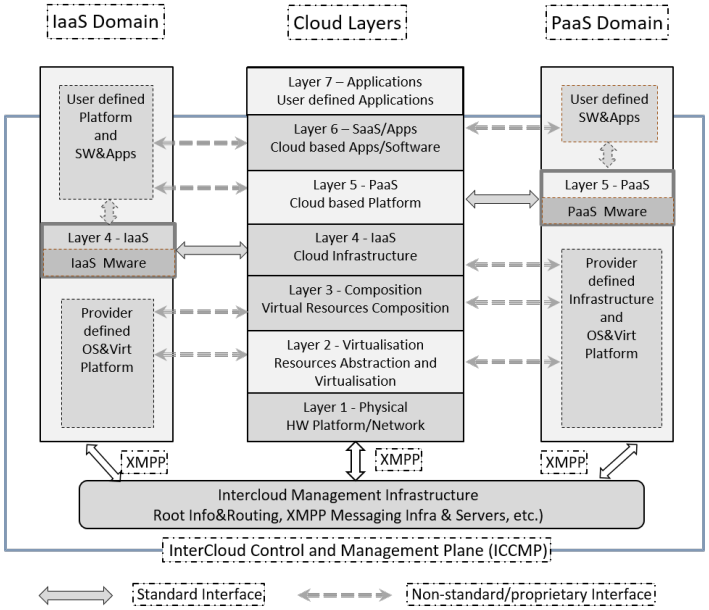

Fig. 2. Example ICCMP communication between the IaaS and the PaaS domains that may use standard interfaces and proprietary interfaces. XMPP can be used for inter-domain multi-layer signaling and control messages.

IV. XMPP Overview

In this section XMPP will be examined more closely to determine its applicability for performing the necessary functions for the ICCMP.

XMPP (Extensible Messaging and Presence Protocol) was initially developed in the late 90's in response to the growing number of incompatible IM (Instant Messaging) platforms that were in general use [6]. Jeremie Miller decided to create an alternative and open communication protocol and released the first version of 'Jabber' in 1999. This protocol was then standardized by the IETF in a number of RFCs, the most current being RFC6120 (XMPP: Core) [7] and RFC6121 (XMPP: Instant Messaging and Presence) [8].

XMPP is distributed in a similar way to SMTP [9]. Each domain will generally have one server representing it. Each server will receive connections from its own local clients as well as maintaining direct connections to other servers with which it wants to communicate. This is the key design difference between XMPP and SMTP.

Although XMPP is generally used for sending instant messages (IM applications), the architecture itself is designed to provide near real-time messaging for any purpose. The core of XMPP already supports key features that the ICCMP would need, such as presence and service discovery. With the Jingle XEP, XMPP can also be used in a similar fashion to SIP, to create, manage and destroy out-of-band connections. This provides, at scale with a trusted technology, all the features needed for the ICCMP.

Some potentially useful services for Intercloud (taken from XMPP: The Definitive Guide [6]):

- Presence (RFC-3921) [10]

- Roster (RFC-3921) [10]
- Notifications (XEP-0060) [11]
- Service Discovery (XEP-0030) [12]
- Jingle (effectively SIP [13] over XMPP) (XEP-0166) [14]

A. XMPP Servers

There is usually one server per domain and it is most commonly hosted by the entity that owns that domain. This draws interesting parallels to the Intercloud. It is likely that each domain of control (cloud provider) will want to run its own Point of Presence (PoP), that is each will want to maintain administrative control over its own node on the XMPP network. This distributed nature, and localization of control, is ideal in the Intercloud environment.

XMPP servers find each other based on their domain name. There is a standard DNS SRV record that an XMPP server will query in order to find which server is authoritative for a given domain. Once the server has been identified, XMPP will establish a connection to the server port specified by DNS. It will then verify the remote server's identity using TLS and certificate-based authentication. This authentication is two-way - the receiving server will also verify the identity of the incoming server. Because the server is trusted, messages coming from that server's domain, will also be trusted as it is assumed that the server is authoritative.

Therefore the XMPP architecture can really be thought of as similar to SMTP. As SMTP is well known and understood in the industry this makes working with and adjusting to XMPP much more tractable.

V. XMPP Services

A. Presence

Presence is one of the most powerful features of native XMPP. The ability to determine whether another user or entity is available and its current status makes real-time messaging more flexible. When a user's status changes, this update is pushed in real time to all interested parties.

B. Roster

The roster provides a list of known and authenticated users that the owner is allowed to see and interact with. When a user signs on, its status is broadcast to every account on its roster. Those accounts will update the user's status and then send their current status in return.

C. Notifications

Notifications are similar to one-to-one messages but the system is designed to push a large number of updates efficiently in one direction. It is also similar in design to a lightweight PubSub system in that notifications are sent to channels or topics. This allows clients to subscribe to specific updates and notifications that interest them.

D. Service Discovery

Service Discovery allows agents to determine what facilities another agent or server supports. This is particularly important in XMPP because of how diverse the landscape is in terms of implementations.

E. Jingle

Jingle is an extension to XMPP (XEP-0166) [14] that implements a negotiation protocol that is heavily inspired by SIP [13]. This allows for very flexible negotiation, creation and destruction of out-of-band connections. This allows XMPP to remain simple but to provide support for advanced integration with other protocols and systems.

VI. Benefits of using XMPP

There are a number of benefits in using XMPP as the underlying communication protocol for Intercloud communication and the ICCMP layer in particular. Some of the benefits include:

- Open standard (IETF / RFC)
- Security
- Scalability
- Real time
- Implementations already exist
- Mature technology
- Documented extension process

A. Open standard

The XMPP protocol has been well documented as an open standard by the IETF. These have been formalized in RFC6120 [7] and RFC6121 [8]. As part of the standardization process, the protocol has been thoroughly reviewed and tested. In addition, as the protocol is an open standard, it can be freely implemented without worrying about licensing or other hindrances.

B. Security

XMPP leverages TLS encryption and certificate verification to provide strong authentication for remote servers. There are a number of different implementations for servers, clients and supporting libraries that allow for a diverse code base. The XMPP standard is still evolving and additional improvements to security are ongoing projects.

C. Scalability

XMPP is highly scalable. As anyone may deploy an XMPP server, like SMTP, there is effectively no limit on how many servers may be deployed. In addition, XMPP uses a 'PUSH' model.

D. Real time

Due to the PUSH model, XMPP users are able to communicate in real time. As long as the receiving agent is logged into their server, they will receive messages as soon as they become available.

E. Implementations already exist

There are a large number of XMPP servers and clients (applications and libraries) available for use. By being able to re-use existing infrastructure, the system will necessarily be layered. Development efforts can then focus on implementing ICCMP without the need to focus on low-level transport details.

F. Mature technology

XMPP has been around for 15 years in one form or another. It has stood the test of time and numerous production systems (such as Google Chat) have been built on top of it. This maturity leads to a great deal of confidence in the architecture which in turn makes adoption an easier and more straight forward process.

G. Documented extension process

XMPP has the concept of XEP's (XMPP Extension Protocols). Enhancements that would enable XMPP to handle Intercloud would not simply be 'hacked' into a system. Instead the approach will be documented, examined by the XMPP community (which is welcoming and supportive) and greatly improved with their feedback. Once an XEP has been approved, any vendor may implement that feature. XMPP features are detectable at run time with XMPP's service discovery.

VII. XMPP and Intercloud

The previous sections introduced the XMPP protocol and highlighted some of the features that might make it a good fit for Intercloud. In addition in the last section, the benefits of XMPP as a choice for a transport layer were also discussed with an emphasis on reliability, scalability and long-term viability. Now that XMPP has been introduced, a brief look at how it might fit into an Intercloud framework can be considered.

A. Payload agnostic

XMPP is based on XML streams. It has the ability to carry any data that can be represented in XML (including binary if it is Base64 encoded). It can theoretically carry any sort of messages between agents in both directions. This allows for great flexibility in the design of the ICCMP where it could then be assumed that the underlying protocol (XMPP) would take care of routing and the delivery of messages. This can then be descoped from the ICCMP layer.

B. Discover Remote Clouds

XMPP uses DNS SRV records to discover which server is responsible for which domain. This means that if an agent wishes to communicate with *Cloud Company A*, it can simply look up the SRV records for their domain. Once an agent is connected to a Cloud, it can use Service Discovery to find other nodes.

C. Service Discovery

An agent can use Service Discovery to determine whether a given node offers 'Cloud Services'. If the facilities are available, the agent can request more information about what is available. This could include which types of service (such as compute or storage) are available or potentially even price lists. This can all be done at the XMPP layer without recourse to a higher protocol.

D. Status Discovery

Assuming that the agent connects to a cloud that it is authorized to use, it could ask for its roster or service list. This would list all of the services belonging to that agent and their current status. The agent would be able to discover the current state of its account with a given cloud provider. Although this example is described from an agent point of view, this agent could easily belong to another cloud rather than an end user.

E. Resource Creation

An agent would be able to send requests to a remote node to instantiate resources. By representing this in XMPP, the remote resource could be modeled as an XMPP identity or account. For example when a remote cloud creates a resource, it can create an entry in its roster. At this stage, the original agent can now see the status of the resource and interact with it directly. This could cycle through different statuses. These would be pushed in real time to all entities on its roster which would include the agent.

The remote agent could be running on the cloud resource itself or it could simply be an abstraction. It can also see its resources and its current status and presence. The agent can interact with it directly and because native XMPP is being used, this could be easily integrated into client libraries. No additional protocol would be needed over the XMPP cloud extensions.

VIII. Raft Implementation

As part of ongoing research on the IEEE P2302 project, a research project was undertaken to take a message-based protocol (in this case the Raft distributed consensus algorithm [15]) and attempt to implement it natively in XMPP. There were two primary goals for this research, namely, to determine how easy it would be to extend XMPP natively to a new protocol and how easy it would be to work with the community's standardization process to have the protocol accepted on the Standards track.

Extending XMPP (where the 'X' stands for extensible) was a straight forward process. As XMPP is a streaming XML protocol, it lends itself to extension. Creating the stanzas to carry the Raft payload was mostly a case of matching the content from Raft to an appropriately named attribute in XMPP. The difficulty then is not in creating the XMPP implementation but in having a well-developed protocol to actually implement in XMPP. XMPP is also amenable to iterative development.

Working with the XSF (XMPP Standards Foundation) to approve the 'Raft over XMPP' XEP (XMPP Extension Proposal) for the Standards track was a good experience that ultimately ended in the creation of 'XEP-0362 Raft Over XMPP' [16]. There are a number of existing XEPs such as XEP-0001 (XMPP Extension Protocols) [17] and XEP-0143 (Guidelines for Authors of XMPP Extension Protocols) [18] which assist in the authoring and preparation of XEPs for consideration by the XSF Council. There is an active mailing list for discussing potential new standards and getting feedback from

the community (standards@xmpp.org), which leads to more robust and higher quality protocols. The XSF Council then votes on proposed XEPs (known informally as ProtoXEPs) to determine if they should be accepted onto the Standards track. In either case, feedback is given to the author(s) on how to improve and further refine their XEP proposal.

In particular then, XMPP is not only easy to extend but explicitly designed to be so. Implementing the Raft consensus algorithm and preparing the XEP, especially with the guidance provided from a very active community, was straight forward. The standardization process itself is easy to understand and presented no particular obstacles, making the implementation of the ICCMP and its standardization by the XSF a very tractable proposition.

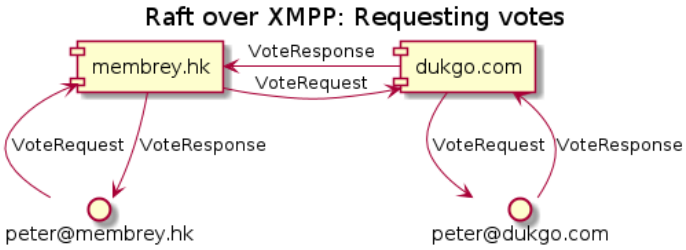

Fig. 3. Raft over XMPP: Voting for a new leader

IX. Conclusion

This paper introduced XMPP and briefly discussed its architecture. It then discussed some of the services that XMPP provides natively that could be put to good use in developing the ICCMP layer (features such as Service Discovery and Presence). Next, a number of additional benefits surrounding the XMPP protocol (such as community, openness and scalability) were highlighted. The question of where XMPP could be applied to Intercloud communication was discussed and some examples were given of how the majority of the ICCMP layer could be implemented directly in XMPP.

The Intercloud needs a reliable means of exchanging information and managing connections. Such a system would need to be scalable, distributed and reliable. XMPP as a base protocol can supply the vast majority of those needs immediately using existing technology. For features specific to Intercloud, like the XEP for Raft (XEP-0362), these features can be easily formalized and put together in an open specification for all parties to use.

XMPP is a strong candidate for the Intercloud ICCMP layer and potentially any protocol that requires near real time messaging. Although originally designed for Instant Messaging in the form of Jabber, when the protocol was standardized by the IETF in RFC's 6120 [7] and RFC 6121 [8], as XMPP, it was with the specific intention of making the protocol easily extensible. These extensions are managed and curated by the XSF (XMPP Standards Foundation).

As discussed in previous sections, XMPP has a number of features built into the protocol that make it amenable to implementing the ICCMP layer. It's an pre-existing protocol that has been used for over 15 years in real world scenarios. It comes with security built in, with SASL authentication and TLS encryption as standard. It supports presence notifications as well as service discovery. In short, many of the key requirements for implementing the ICCMP are immediately available in XMPP for use.

However XMPP does have its limitations. It is a streaming XML protocol which makes it a poor choice for carrying large amounts of binary data. As discussed early, the Jingle XEP implements support for out-of-band file transfers, so although XMPP is not necessarily an ideal carrier, it comes with built-in support for signaling and managing external connections.

X. Acknowledgements

This research involved collaboration with the IEEE Intercloud Testbed Project Group. The related Intercloud research was supported by EU Projects CYCLONE and GEANT4 (grant agreement no. 691567, Research and Education Networking), and CYCLONE (grant agreement no. 644925, Advanced Cloud Infrastructures and Services).

References

[1] Y. Demchenko, M. X. Makkes, R. Strijkers, and C. De Laat, "Intercloud architecture for interoperability and integration," in *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*. IEEE, 2012, pp. 666–674.

[2] L. Badger, T. Grance, R. Patt-Corner, and J. Voas, "Draft cloud computing synopsis and recommendations," *NIST special publication*, vol. 800, p. 146, 2011.

[3] Y. Demchenko, C. Ngo, C. De Laat, J. Rodriguez, L. M. Contreras, J. A. Garcia-Espin, S. Figuerola, G. Landi, and N. Ciulli, "Intercloud architecture framework for heterogeneous cloud based infrastructure services provisioning on-demand," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. IEEE, 2013, pp. 777–784.

[4] D. Bernstein and D. Vij, "Intercloud exchanges and roots topology and trust blueprint," in *Proc. of 11th International Conference on Internet Computing*, 2011, pp. 135–141.

[5] P. V. Mockapetris, "Domain names-concepts and facilities," 1987.

[6] P. Saint-Andre, K. Smith, and R. Tronçon, *XMPP: the definitive guide*. " O'Reilly Media, Inc.", 2009.

[7] P. Saint-Andre, "Rfc 6120: Extensible messaging and presence protocol (xmpp): Core (2011)," *URL http://tools. ietf. org/html/rfc6120 [2011-08-28]*.

[8] ——, "Extensible messaging and presence protocol (xmpp): Instant messaging and presence (rfc 6121), ietf, march 2011."

[9] J. Klensin, "Rfc 5321simple mail transfer protocol (smtp)," RFC 5321, Tech. Rep., 2008.

[10] The IETF, "RFC3921 – Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence," Tech. Rep., 2004.

[11] P. Millard, P. Saint-Andre, and R. Meijer, "Xep-0060: Publish-subscribe," *XMPP Standards Foundation*, vol. 1, p. 13, 2010.

[12] J. Hildebrand, P. Millard, R. Eatmon, and P. Saint-Andre, "Xep-0030: service discovery," *XMPP Standards Foundation, Tech. Rep*, 2008.

[13] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "Sip: session initiation protocol," Tech. Rep., 2002.

[14] S. Ludwig, J. Beda, P. Saint-Andre, R. McQueen, S. Egan, and J. Hildebrand, "Xep-0166: Jingle," *XMPP Standards Foundation*, 2009.

[15] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm (extended version)," 2014.

[16] P. Membrey, "Xep-0362: Raft over xmpp," *XMPP Standards Foundation*, 2015.

[17] P. Saint-Andre, "Xep-0001: Xmpp extension protocols," *Viitattu*, vol. 27, p. 2013, 2010.

[18] ——, "Xep-0134: Xmpp design guidelines," 2010.